UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/632,216	07/31/2003	Gerard Chauvel	TI-35445	1110

23494 7590 02/25/2008
TEXAS INSTRUMENTS INCORPORATED
P O BOX 655474, M/S 3999
DALLAS, TX 75265

EXAMINER	
PETRANEK, JACOB ANDREW	
ART UNIT	PAPER NUMBER
2183	

NOTIFICATION DATE	DELIVERY MODE
02/25/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

uspto@ti.com
uspto@dlemail.itg.ti.com

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/632,216
Filing Date: July 31, 2003
Appellant(s): CHAUVEL ET AL.

Utpal D. Shah, Reg. No. 60,047
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 12/14/2007 appealing from the Office action mailed 8/17/2007.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

No amendment after final has been filed.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

Zaidi (U.S. 6,581,154), Seal et al. (U.S. 6,965,984), Gee et al. (U.S. 6,317,872), and Greenberger et al. (U.S. 6,092,179) are relied upon as evidence.

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Maintained Claim Rejections - 35 USC § 112

1. The following is a quotation of the first paragraph of 35 U.S.C. 112:

> The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same and shall set forth the best mode contemplated by the inventor of carrying out his invention.

2. Claim 10-14 are rejected under 35 U.S.C. §112, first paragraph, as containing subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and/or use the invention.

Claim 10 recites the limitation "the determining independent of the type of the instruction." In paragraphs 17-18 of the specification, regular java bytecodes and complex bytecodes are described. A simple bytecode is an instruction that can execute in a single or several cycles. A complex bytecode takes longer and may be replaced by a micro-sequence of other instructions. Paragraph 26 of the specification describes using a micro-sequence vector table to store all of the bytecodes and use a bit to determine if the bytecode will be replaced by a micro-sequence. A type of the instruction is described in paragraph 25 of the specification as being determined upon decoding. Thus, a type of the instruction is interpreted as including parts of the instruction that identify what and how the instruction is to operate. When initially loading the micro-sequence vector table, the type of the instruction must be taken into consideration when determining which bytecodes will be replaced by a micro-sequence

and what the address will be to point to this sequence of instructions. Thus, when later accessing the micro-sequence vector table, the type of instruction determines if the instruction is directly executed or replaced by a micro-sequence. The specification doesn't enable one of ordinary skill in the art as to how determining which bytecodes will be replaced doesn't take into consideration the form of the instruction without causing undue burden and/or experimentation. For examination purposes, the limitation will be interpreted as the determination is dependent on the instruction.

3. Claims 10-14 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement. The claim(s) contains subject matter which was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor(s), at the time the application was filed, had possession of the claimed invention.

Claim 10 recites the limitation "the determining independent of the type of the instruction." In paragraphs 17-18 of the specification, regular java bytecodes and complex bytecodes are described. A simple bytecode is an instruction that can execute in a single or several cycles. A complex bytecode takes longer and may be replaced by a micro-sequence of other instructions. Paragraph 26 of the specification describes using a micro-sequence vector table to store all of the bytecodes and use a bit to determine if the bytecode will be replaced by a micro-sequence. A type of the instruction is described in paragraph 25 of the specification as being determined upon decoding. Thus, a type of the instruction is interpreted as including parts of the instruction that identify what and how the instruction is to operate. When initially loading

the micro-sequence vector table, the type of the instruction must be taken into consideration when determining which bytecodes will be replaced by a micro-sequence and what the address will be to point to this sequence of instructions. Thus, when later accessing the micro-sequence vector table, the type of instruction determines if the instruction is directly executed or replaced by a micro-sequence. Therefore, there isn't any support within the specification at the time the application was filed to support the additional limitation.

4. Claims 11-14 are rejected due to their dependency.

Maintained Claim Rejections - 35 USC § 102

5. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in-
> (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effect under this subsection of a national application published under section 122(b) only if the international application designating the United States was published under Article 21(2)(a) of such treaty in the English language; or
> (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that a patent shall not be deemed filed in the United States for the purposes of this subsection based on the filing of an international application filed under the treaty defined in section 351(a).

6. Claims 10 and 13 are rejected under 35 U.S.C. §102(e) as being anticipated by Gee (U.S. 6,317,872).

7. As per claim 10:

Gee disclosed a method comprising:

Fetching an instruction (Gee: Figure 1 element 104, column 8 lines 63-67)(The instructions are fetched from element 104); and

Determining whether said instruction is to be executed or replaced by a group of other instructions, the determining dependent of the type of instruction (Gee: Figure 2 element 200, column 9 lines 5-52)(The control store contains the micro sequence for each macro java bytecode and is used to replace the bytecode. All types of instructions are replaced with a group of instructions.).

8. As per claim 13:

Gee disclosed the method of claim 10 further including switching an active program counter between two program counters when replacing the instruction with the group of instructions (Gee: Figure 2 element 204 and 226, column 8 lines 49-67 continued to column 9 lines 1-52)(The primary program counter, element 204, is functioning to fetch java bytecodes and the micro program counter, element 226, is selected to fetch and execute the micro program. Thus, having the same functionality.).

Maintained Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. §103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 10-12, 14, and 22-23 are rejected under 35 U.S.C. §103(a) as being unpatentable over Zaidi (U.S. 6,581,154).

11. As per claim 10:

Zaidi disclosed a method comprising:

Fetching an instruction (Zaidi: Figure 2 element 205, column 3 lines 55-67 continued to column 4 lines 1-7)(Instructions are fetched by macroinstructions from the uROM.); and

Determining whether said instruction is to be executed or replaced by a group of other instructions, the determining dependent of the type of instruction (Zaidi: Figure 2 elements 203, 205, and 210, column 3 lines 55-67 continued to column 4 lines 1-7)(The micro instruction sequencer translates macroinstructions into Uops and Suops. The instructions stored in the uROM are executed or replaced depending on if they are Uops or Suops. It's obvious to one of ordinary skill in the art that after the instructions are fetched from the uROM, there must be a determining step that determines if the instruction fetched is a Uop that needs to be sent directly to be executed or if the instruction fetched is a Suop that needs to be replaced by a other instructions in the expander element. The type of instruction being analyzed, i.e. Suop of Uop, directly determines if the instruction is executed or replaced with a sequence of instructions.).

12. As per claim 11:

Zaidi disclosed the method of claim 10 further including replacing the instruction with said group of other instructions (Zaidi: Figure 2 element 210, column 3 lines 55-67 continued to column 4 lines 1-7)(The micro instruction sequencer logic determines if the

instruction is a Uop or a Suop that will further be expanded and replaced by a group of instructions.).

13. As per claim 12:

Zaidi disclosed the method of claim 10 wherein determining whether the instruction is to be executed or replaced includes determining a value of a bit associated with the instruction (Zaidi: Figure 2 element 205, column 4 lines 1-7)(Element 205 stores both Uops and Suops. The S U ops are to be further expanded and the bits associated with them indicate that they are to be replaced by a group of instruction by element 210. The Uops have bits associated with them that indicate no further expansion is needed and that they can be sent directly to the dispatch queue. Thus having the same functionality.).

14. As per claim 14:

Zaidi disclosed the method of claim 10 further including programming a table to specify which instructions are to be executed directly and which instructions are to be replaced by a group of instructions (Zaidi: Figure 2 element 205, column 4 lines 1-7)(Element 205 stores both Uops and Suops. The Suops are to be further expanded and the bits associated with them indicate that they are to be replaced by a group of instruction by element 210. The Uops have bits associated with them that indicate no further expansion is needed and that they can be sent directly to the dispatch queue. Thus having the same functionality.).

15. As per claim 22:

Zaidi disclosed a processor, comprising:

Decode logic that decodes instructions (Zaidi: Figure 2 elements 203, 205, and 210, column 3 lines 55-67 continued to column 4 lines 1-7)(It's obvious to one of ordinary skill in the art that the Uops and Suops retrieved from the uROM are decoded to determine if they are sent to be directly executed or to be expanded within element 200.); and

A means for determining whether an instruction is to be executed or replaced by a micro-sequence of other instructions (Zaidi: Figure 2 elements 203, 205, and 210, column 3 lines 55-67 continued to column 4 lines 1-7)(The micro instruction sequencer translates macroinstructions into Uops and Suops. The instructions stored in the uROM are executed or replaced depending on if they are Uops or Suops. It's obvious to one of ordinary skill in the art that after the instructions are fetched from the uROM, there must be a determining step that determines if the instruction fetched is a Uop that needs to be sent directly to be executed or if the instruction fetched is a Suop that needs to be replaced by a other instructions in the expander element.).

16. As per claim 23:

Zaidi disclosed the processor of claim 22 further including a means for replacing the instruction with the micro-sequence (Zaidi: Figure 2 element 203, column 3 lines 55-67 continued to column 4 lines 1-7)(The micro instruction sequencer logic determines if the instruction is a Uop or a S Uop that will further be expanded and replaced by a group of instructions.).

17. Claims 18 and 21 are rejected under 35 U.S.C. §103(a) as being unpatentable over Seal et al. (U.S. 6,965,984).

18. As per claim 18:

Seal disclosed an electronic device, comprising:

Decode logic that decodes instructions, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different from the first instruction set (Seal: Figure 1 element 6, column 5 lines 60-67 continued to column 6 lines 1-9)(It's obvious to one of ordinary skill in the art at the time of the invention that there would have to be some sort of predecoding done to initially detect java bytecodes to send to element 6 or detect ARM opcodes that will be bypassed around element 6. The first and second instruction sets are the java bytecodes and the ARM opcodes, which are different from each other.); and

A vector table comprising a plurality of entries, each entry corresponding to a separate instruction and including a first field indicating whether the corresponding instruction is to be executed by the electronic device or whether the instruction is to be replaced by a predetermined group of instructions stored in memory (Seal: Figure 2 element 24, column 6 lines 60-67 continued to column 7 lines 1-23)(All of the entries in element 24 store pointers to a group of instructions that will replace the instruction. Thus having the same functionality.).

19. As per claim 21:

Seal disclosed the electronic device of claim 18 wherein the group of instructions terminates with a predetermined instruction (Seal: Figure 2 element 26, column 7 lines

24-40)(The BXJ instruction will cause a group of instructions to terminate. Thus having the same functionality.).

20. Claims 1-4, 7-9, and 19-20 are rejected under 35 U.S.C. §103(a) as being unpatentable over Seal et al. (U.S. 6,965,984), in view of Gee et al. (U.S. 6,317,872).

21. As per claim 1:

Seal disclosed a processor comprising:

Fetch logic that retrieves instructions from memory (It's obvious to one of ordinary skill in the art that fetch logic exists to fetch the Java bytecodes and ARM bytecodes.);

Decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set (Seal: Figure 1 element 6, column 5 lines 60-67 continued to column 6 lines 1-9)(It's obvious to one of ordinary skill in the art at the time of the invention that there would have to be some sort of predecoding done to initially detect java bytecodes to send to element 6 or detect ARM opcodes that will be bypassed around element 6. The first and second instruction sets are the java bytecodes and the ARM opcodes, which are different from each other.);

Wherein an instruction of the first instruction set is replaced by a micro-sequence comprising one or more instructions of the second instruction set (Seal: Figure 2 element 24, column 6 lines 60-67 continued to column 7 lines 1-23)(All of the entries in element 24 store pointers to a group of instructions that will replace the instruction.

Thus having the same functionality. The first instruction set is the Java bytecodes and the second is the ARM opcodes.).

Seal failed to teach an active program counter selected as either a first program counter or a second program counter and the active program counter switches between the first and second program counters.

However, Gee disclosed an active program counter selected as either a first program counter or a second program counter (Gee: Figure 2 element 204 and 226, column 8 lines 49-67 continued to column 9 lines 1-52)(The primary program counter, element 204, is functioning to fetch java bytecodes and the micro program counter, element 226, is selected to fetch and execute the micro program. Thus having the same functionality. When combined with Seal, the primary program counter would also work with the ARM opcodes that are normally fetched. The micro program counter would be used with the plurality of ARM opcodes that a single Java bytecode is translated into.); and

The active program counter switches between the first and second program counters (Gee: Figure 2 element 204 and 226, column 8 lines 49-67 continued to column 9 lines 1-52)(The primary program counter, element 204, is functioning to fetch java bytecodes and the micro program counter, element 226, is used to fetch and execute the micro program. Thus having the same functionality.).

Seal disclosed a method of translating a single java bytecode into a plurality of ARM opcodes through using a table and address pointers. Seal failed to teach how to differentiate the microinstructions from one another when they are executed because

they all will share the same program counter from the initial fetched java bytecode. One of ordinary skill in the art would have been motivated by this to find Gee using a micro-program counter for microcode instructions generated from macroinstructions. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the two program counters for the advantage of allowing for translated java bytecodes into a plurality of ARM opcodes to be tracked through the processor.

22. As per claim 2:

Seal and Gee disclosed the processor of claim 1, including a vector table accessible by said decode logic, said vector table including information which specifies whether an instruction is to be replaced by a micro-sequence (Seal: Figure 2 element 24, column 6 lines 60-67 continued to column 7 lines 1-23)(All of the entries in element 24 store pointers to a group of instructions that will replace the instruction. Thus having the same functionality.).

23. As per claim 3:

Seal and Gee disclosed the processor of claim 2 wherein the information is provided to the vector table from a block of memory accessible to the processor by an indirect addressing mode used in a repeat loop comprising at least one instruction (Seal: Figure 10, column 10 lines 34-67 continued to column 11 lines 1-20)(The vector table is initialized with value in a repeat loop shown in figure 10.).

24. As per claim 4:

Seal and Gee disclosed the processor of claim 2 wherein the vector table comprises a plurality of entries and any one entry can be modified independently of the

other entries (Seal: Figure 10 element 120, column 10 lines 34-67 continued to column 11 lines 1-20)(Upon initialization, each entry is written independently of each other. Thus having the same functionality.).

25. As per claim 7:

Seal and Gee disclosed the processor of claim 1 wherein the active program counter again switches between the first and second program counters when the micro-sequence is completed (Gee: Figure 2 element 204 and 226, column 8 lines 49-67 continued to column 9 lines 1-52)(Upon the completion of the micro program, another macro java bytecode is fetched to execute. With the micro PC not fetching and executing the micro program, the PC will fetch additional java bytecodes as the active PC.).

26. As per claim 8:

Seal and Gee disclosed the processor of claim 1 wherein the second program counter is used to fetch and decode instructions comprising a micro-sequence and switching between the first and second program counters comprises switching from the first program counter to the second program counter and loading the second program counter with a starting address of the micro-sequence (Gee: Figure 2 elements 204 and 226, column 8 lines 49-67 continued to column 9 lines 1-52)(Figure two shows that the micro PC is used to index into the micro instruction storage to fetch instructions. It's inherent then that the micro PC is given the starting address of the micro sequence of instructions to correctly fetch and execute them.).

27. As per claim 9:

Seal and Gee disclosed the processor of claim 1 wherein a plurality of instructions are replaceable by a corresponding micro-sequence (Gee: Column 8 lines 50-62)(The macro java bytecodes are replaced by a pointer that points to a sequence of micro instructions.).

28. As per claim 19:

Seal disclosed the electronic device of claim 18.

Seal failed to teach further including an active program counter selected as either a first program counter or a second program counter, wherein an instruction is replaced by the group of instructions and the active program counter concurrently switches from the first to the second program counter.

However, Gee disclosed further including an active program counter selected as either a first program counter or a second program counter, wherein an instruction is replaced by the group of instructions and the active program counter concurrently switches from the first to the second program counter (Gee: Figure 2 element 204 and 226, column 8 lines 49-67 continued to column 9 lines 1-52)(The primary program counter, element 204, is functioning to fetch java bytecodes and the micro program counter, element 226, is selected to fetch and execute the micro program. Thus having the same functionality. The combination of Seal and Gee results in the micro program counter being used for when a plurality of ARM opcodes are generated for a single java bytecode.).

Seal disclosed a method of translating a single java bytecode into a plurality of ARM opcodes through using a table and address pointers. Seal failed to teach how to

differentiate the micro-instructions from one another when they are executed because they all will share the same program counter from the initial fetched java bytecode. One of ordinary skill in the art would have been motivated by this to find Gee using a micro-program counter for microcode instructions generated from macroinstructions. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the two program counters for the advantage of allowing for translated java bytecodes into a plurality of ARM opcodes to be tracked through the processor.

29. As per claim 20:

Seal disclosed the electronic device of claim 18.

Seal failed to teach wherein upon switching the active program counter, the first program counter is incremented.

However, Gee disclosed wherein upon switching the active program counter, the first program counter is incremented (Gee: Figure 2 elements 204 and 236, column 8 lines 49-67 continued to column 9 lines 1-52)(When another macro instruction is to be fetched from memory, the PC is incremented to insure that the correct instruction is fetched instead of an instruction previously fetched. Thus having the same functionality.).

Seal disclosed a method of translating a single java bytecode into a plurality of ARM opcodes through using a table and address pointers. Seal failed to teach how to differentiate the micro-instructions from one another when they are executed because they all will share the same program counter from the initial fetched java bytecode. One of ordinary skill in the art would have been motivated by this to find Gee using a micro-

program counter for microcode instructions generated from macroinstructions. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement the two program counters for the advantage of allowing for translated java bytecodes into a plurality of ARM opcodes to be tracked through the processor.

30. Claims 5-6 are rejected under 35 U.S.C. §103(a) as being unpatentable over Seal et al. (U.S. 6,965,984), in view of Gee et al. (U.S. 6,317,872), further in view of Zaidi (U.S. 6,581,154).

31. As per claim 5:

Seal and Gee disclosed the processor of claim 1.

Seal and Gee failed to teach including a micro-sequence vector table comprising a plurality of entries, each entry corresponding to a separate instruction and associated with a bit indicating whether the corresponding instruction is to be executed by the processor or whether the instruction is to be replaced by a micro-sequence.

However, Zaidi disclosed including a micro-sequence vector table comprising a plurality of entries, each entry corresponding to a separate instruction and associated with a bit indicating whether the corresponding instruction is to be executed by the processor or whether the instruction is to be replaced by a micro-sequence (Zaidi: Figure 2 element 205, column 4 lines 1-7)(Element 205 stores both U ops and S U ops. The S U ops are to be further expanded and the bits associated with them indicate that they are to be replaced by a group of instruction by element 210. The U ops have bits associated with them that indicate no further expansion is needed and that they can be

sent directly to the dispatch queue. Thus having the same functionality.).

Gee disclosed a processor that replaces macro java bytecodes with a micro sequence of instructions. Gee disclosed that a java bytecode is essentially a pointer to a sequence of microinstructions (Gee: Column 8 lines 57-59). However, Gee doesn't disclose the process of the replacement. One of ordinary skill in the art would have been motivated by the lack of the disclosure on the replacement process to find additional details on the process. Zaidi disclosed using a vector table of entries that will replace a macroinstruction with microinstructions. Thus, it would have been obvious to one of ordinary skill in the art at the time of the invention to implement a vector table to contain pointers to the microinstructions that will replace the java bytecodes.

32. As per claim 6:

Seal, Gee, and Zaidi disclosed the processor of claim 5 wherein at least some of the entries include a reference to a memory location in which a micro-sequence is stored depending if the associated bit indicates that the instruction is to be replaced by a micro-sequence (Zaidi: Figure 2 elements 205 and 210, column 4 lines 1-7)(Element 205 inherently includes information about a memory location that stores the instructions to be replaced.).

33. Claims 15-17 are rejected under 35 U.S.C. §103(a) as being unpatentable over Seal et al. (U.S. 6,965,984), in view of Gee et al. (U.S. 6,317,872), and in view of Greenberger et al. (U.S. 6,092,179).

34. As per claim 15:

Claim 15 essentially recites the same limitations of claim 1. Claim 15 additionally recites the following limitations:

Seal and Gee additionally disclosed a second processor (Gee: Figure 1 element 100).

Seal and Gee failed to teach a first processor coupled to the second.

However, Greenberger disclosed a first processor coupled to the second (Greenberger: Figure 2 elements 2 and 7, column 3 lines 49-67).

One technique to add new functionality to a processor is to add a co-processor with the added technique (Greenberger: Column 2 lines 1-20). One of ordinary skill in the art would have been motivated to add the combine the processor of Greenberger and Gee for the added functionality of executing macro java bytecodes and replacing them with a micro sequence. Thus, it would have been obvious to one of ordinary skill in the art to combine the two processors for the added functionality of executing java bytecodes.

35. As per claim 16:

Seal, Gee, and Greenberger disclosed the system of claim 15, wherein said second processor further includes a micro-sequence vector table comprising a plurality of entries, each entry corresponding to a separate instruction and including a field that indicates whether the corresponding instruction is to be executed by the second processor or whether the instruction is to be replaced by a micro-sequence (Seal: Figure 2 element 24, column 6 lines 60-67 continued to column 7 lines 1-23)(All of the entries in element 24 store pointers to a group of instructions that will replace the

instruction. Thus having the same functionality.).

36. As per claim 17:

Claim 17 essentially recites the same limitations of claim 6. Therefore, claim 17 is rejected for the same reasons as claim 6.

(10) Response to Argument

37. Regarding claims 10-14 rejected under 35 U.S.C. §112, first paragraph, as containing subject matter which was not described in the specification in such a way as to enable one skilled in the art to which it pertains, or with which it is most nearly connected, to make and/or use the invention:

A.) Appellant argues "The specification discloses that the decode logic fetches a Bytecode instruction from an instruction storage, then the decode logic uses the Bytecode as an index into the vector table to examine the associated bit. Thus, the determination whether the bytecode is to be replaced is made independent of the type of Bytecode fetched from the memory."

The examiner disagrees for the following reasons. Many factors are to be considered when determining whether there is sufficient evident to support a determination that a disclosure does not satisfy the enablement requirement and whether any necessary experimentation is "undue". [In re Wands, 858 F.2d 731, 737, 8 USPQ2d 1400, 1404 (Fed. Cir. 1988). *See also MPEP § 2164.01(a).]* For example, the factors include but are not limited to:

(A) The breadth of the claims;

(B) The nature of the invention;

(C) The state of the prior art;

(D) The level of one of ordinary skill;

(E) The level of predictability in the art;

(F) The amount of direction provided by the inventor;

(G) The existence of working examples; and

(H) The quantity of experimentation needed to make or use the invention based on the content of the disclosure

The nature of the invention is determining if a decoded instruction can be directly executed or if it is to be replaced by a micro-sequence of instructions, the determining independent of the type of instruction. The type of the instruction being interpreted as including parts of the instruction that identify what and how the instruction is to operate. The disclosure clearly supports that the determination step of the claim occurs when checking bit 168. When initializing bit 168, the bit is set or clearly directly dependent on the type of instruction. Therefore, a determination of if an instruction is to be directly executed or replaced with a sequence of instructions is based directly on bit 168 being set or cleared, which is directly determined by the type of the instruction.

The state of the prior art shows that determining if a decoded instruction can be directly executed or if it is to be replaced by a micro-sequence of instructions is well-known. However, the prior art shows that the determining step is dependent on the type of instruction to be executed or replaced.

The amount of direction provided by the inventor is not sufficient to allow one of ordinary skill in the art to make or use the invention without undue experimentation. The entire disclosure for the claimed limitation "determining whether said instruction is to be executed or replaced by a group of other instructions, the determining independent of the type of the instruction" is found on page 11 paragraph 25 lines 5-8 and page 13 paragraph 30 lines 1-8 of the specification respectfully and copied below.

> "In general and as described above, the decode logic 152 receives instructions (e.g., instructions 170) from instruction storage 130 via instruction fetch logic 154 (Figure 2) and decodes the instructions to determine the type of instruction for subsequent processing and execution."

> "In operation, the decode logic 152 uses a Bytecode from instructions 170 as an index into micro-sequence vector table 162. Once the decode logic 152 locates the indexed entry 164, the decode logic 152 examines the associated bit 168 to determine whether the Bytecode is to be replaced by a micro-sequence. If the bit 168 indicates that the Bytecode can be directly processed and executed by the JSM, then the instruction is so executed. If, however, the bit 168 indicates that the Bytecode is to be replaced by a micro-sequence, then the decode logic 152 preferably changes this instruction into a "NOP" and sets the micro-sequence-active bit (described above) in the status register R15."

Thus, the direction provided by the inventor only supports a determining step being based on checking bit 168. Checking bit 168 determines if the instruction is to be executed or replaced by a group of other instructions. Since bit 168 is set or cleared directly dependent on the type of the instruction, the determining step is also made directly dependent on the type of the instruction.

There is no existence of working examples of a determining step in the specification that is independent of the type of the instruction. The close working example is shown in figure 4 element 168, which shows that the determination step is directly dependent on the type of instruction. This is the exact opposite of the claimed

invention, which states "determining whether the instruction is to be executed or replaced by a group of other instructions, the determining independent of the type of instruction."

Therefore, since the disclose shows that the determining is dependent upon the type of instruction, the specification fails to enable one of ordinary skill that the determining step is independent of the type of instruction due to the many factors outlined above.

B.) Appellant argues “A single bit, such as element 168, in a table or the table by itself is merely data and imparts no functionality.”

The examiner partially agrees for the following reasons. Of course element 168 is data, as it’s a bit within a data structure of the processor. However, it clearly has a function. Its function is to tell the processor to replace a given instruction with a sequence or not.

38. Regarding claims 10-14 are rejected under 35 U.S.C. 112, first paragraph, as failing to comply with the written description requirement:

Appellant argues “Thus, the specification discloses that the decode logic fetches Bytecode instructions from an instruction storage, then the decode logic uses a Bytecode as an index into the vector table to examine the associated bit. Thus, the determination whether the Bytecode is to be replaced is made independent of the type of Bytecode fetched from the memory. In fact, the determination of the type of Bytecode fetched is only made after reading an entry in the vector table that indicates Bytecode should be directly executed. ”

The examiner disagrees for the following reasons. The Bytecode is used to index into the vector table to examine the associated bit, the examining being the determining step of the claimed limitation. However, the bytecode is originally set directly dependent on the type of instruction that it's being set for. For example, if an add instruction being added to the vector table is to be replaced by a sequence of 3 instructions, bit element 168 is set to replace the add instruction based on its type. When the vector table and element 168 are later accessed, determining if the instruction is to be replaced is directly dependent on the type of instruction accessing the table. Thus, there's no written description support for the claimed limitation.

39. Regarding claims 1-4, 7-9, and 19-20 rejected under 35 U.S.C. §103(a) as being unpatentable over Seal et al. (U.S. 6,965,984), in view of Gee et al. (U.S. 6,317,872):

A.) Appellant argues "Seal teaches that ARM opcodes may be directly supplied to the ARM opcode decoder 10 by bypassing the bytecode translation hardware 6. Thus, Seal appears to teach a decoder 10 that can decode only ARM instructions and a Java bytecode translation hardware 6 that can only translate bytecodes ... the references still fail to teach or fairly suggest a "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set"" for claim 1.

The examiner agrees with the appellant for the following reasons. The appellant is correct in stating that the decoder element 10 in figure 1 only decodes ARM opcodes and not Java bytecodes. However, this is not used by the examiner to reject the claims

and therefore isn't relevant. See part B below for how the examiner rejected the claimed limitation and why the rejection should be upheld.

B.) Appellant argues "Seal teaches that register 19 includes a flag 21 that indicates active or inactive bytecode translation hardware. The flag 21 in register 19 is set by a separate software control and based on the flag 21, the system is provided either Java bytecodes or ARM instructions, but not a mixture. Thus, predecoding is not necessary" for claim 1.

The examiner disagrees for the following reasons. Appellant is correct in stating flag 21 in register 19 indicates whether the data processing system is currently executing JAVA or ARM instructions. However, Seal doesn't disclose how flag 21 is set or cleared. Even if there isn't a mixture of Java and ARM instructions, there will inherently be transitions from a JAVA program to an ARM program, or vice versa, executed by the processor. This transition inherently needs to be detected by the processor to either stop translating JAVA bytecodes or start translating them.

The recent Supreme Court decision of KSR International Co. v. Teleflex Inc. allows for a plurality of different rationales to reject claims. One such rationale is that the claim would have been obvious because "a person of ordinary skill has good reason to pursue the known options within his or her technical grasp. If this leads to the anticipated success, it is likely the product is not of innovation but of ordinary skill and common sense." The problem one of ordinary skill in the art faces is that there inherently will be a transition from JAVA bytecodes to ARM instructions, or vice versa, on the processor of Seal. This transition must be detected, but Seal doesn't disclose

how this transition is detected. One of ordinary skill in the art would realize that there had been a finite number of identified predictable potential solutions. Predecoding the incoming instructions or bytecodes prior to either the JAVA bytecodes being received by the translation hardware or the ARM instructions being received by the decoder would be one of the finite solutions. This allows for detecting JAVA bytecodes in order to set flag 21 or detecting ARM instructions to clear flag 21 in the processor of Seal. Thus, one of ordinary skill in the art could have pursed the known method of predecoding to detect this transition with a reasonable expectation of success.

C.) Appellant argues "Thus, Seal appears to teach a table of pointers pointing to ARM code fragments, but the table does not include information which indicates whether the JAVA bytecode is to be replaced by the ARM code fragments … Moreover, Seal teaches that the table is pointers is accessible by the Java Translation hardware 6 and not by the ARM opcode decoder 10" for claim 2.

The examiner disagrees for the following reason. The pointer itself in Seal inherently indicates that the JAVA bytecode is to be replaced with an ARM code fragment. The vector table is accessible to the predecoding logic added to Seal in claim 1 because the predecoding occurs prior to accessing the vector table. This is the same setup that occurs in applicant's figure 4, where instructions are first decoded and then access the microsequence vector table.

40. Regarding claims 10 and 13 rejected under 35 U.S.C. §102(e) as being anticipated by Gee (U.S. 6,317,872):

Appellant argues "Gee teaches that each and every bytecode points to a sequence of microinstructions that are executed in place of the bytecode. Thus, Gee fails to inherently or expressly teach "determining whether said instruction is to be executed or replaced by a group of other instructions.""

The examiner disagrees for the following reasons. The appellant is correct in summarizing Gee that every bytecode is replaced by a sequence of microinstructions. However, Gee disclosed that the processor makes a determination that all macroinstructions are to be replaced by a sequence of microinstructions instead of directly executing the macroinstructions. Thus, Gee disclosed the claimed limitation.

41. Regarding claims 10-12, 14, and 22-23 rejected under 35 U.S.C. §103(a) as being unpatentable over Zaidi (U.S. 6,581,154):

Appellant argues "Zaidi appears to teach that Suops are merely a way of representing multiple uops. It follows that a Suop is not an instruction that can be executed; rather, a Suop merely is a representation of multiple instructions (uops) … Thus, Zaidi fails to teach or fairly suggest ... fetching an instruction and determining whether said instruction is to be executed or replaced by a group of other instructions"

The examiner disagrees for the following reasons. It is immaterial whether a Suop can be executed or not, as there is no requirement in the claims that the instruction fetched that is replaced with a sequence of instructions must itself be capable of execution on the processor. The Suop is an instruction that is to be replaced by a sequence of instructions. A Suop instruction is still being fetched from the uROM,

element 205 in figure 2. Thus, since a Suop is a fetched instruction and is one to be replaced by a sequence of Uops, Zaidi reads upon the claimed limitation.

42. Regarding claims 15-17 rejected under 35 U.S.C. §103(a) as being unpatentable over Seal et al. (U.S. 6,965,984), in view of Gee et al. (U.S. 6,317,872), and in view of Greenberger et al. (U.S. 6,092,179):

A.) Appellant argues "Seal teaches that ARM opcodes may be directly supplied to the ARM opcode decoder 10 by bypassing the bytecode translation hardware 6. Thus, Seal appears to teach a decoder 10 that can decode only ARM instructions and a Java bytecode translation hardware 6 that can only translate bytecodes ... the reference still fail to teach or fairly suggest a "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set"" for claim 15.

The examiner partially agrees for the following reasons. The examiner agrees with the appellant's assessment of Seal. However, the appellant has ignored the examiner's actual rejection for the claimed limitation. The examiner stated that it's obvious to one of ordinary skill in the art at the time of the invention that there would have to be some sort of predecoding done to initially detect java bytecodes to send to element 6 or detect ARM opcodes that will be bypassed around element 6. Thus, the appellant hasn't properly traversed the rejection of claim 18 and the examiner believes it therefore must be upheld.

B.) Appellant argues "Seal appears to teach a table of pointers accessible by the Java translation hardware 6; however, the table of pointers does not include information which indicates whether the Java bytecodes are to be replaced by the ARM code fragments ... the references still fail to teach of fairly suggest ... each entry corresponding to a separate instruction and including a field that indicates whether the corresponding instruction is to be executed by the second processor or whether the instruction is to be replaced by a micro-sequence" for claim 16.

The examiner disagrees for the following reasons. First, the appellant appears to be implying that the table of pointers, element 24 in figure 2, has multiple entries for an instruction by stating Seal doesn't teach the limitation "each entry corresponding to a separate instruction." There is no support provided by the appellant for such an assertion that there are multiple instructions within the table of pointers for a single instruction. One of ordinary skill in the art would realize that having said multiple entries would only increase the size of the table of pointers and the cost of it. Thus, it's obvious to one of ordinary skill in the art that Seal disclosed a table of pointers with only a single pointer per instruction to be replaced with an ARM code fragment.

The appellant also states that the vector table doesn't contain a field that indicates if an instruction is to be executed or replace by a micro-sequence. However, the pointer itself in Seal is a field that inherently indicates that the JAVA bytecode is to be replaced with an ARM code fragment. The vector table is accessible to the predecoding logic added to Seal in claim 1 because the predecoding occurs prior to

accessing the vector table. This is the same setup that occurs in applicant's figure 4, where instructions are first decoded and then access the microsequence vector table.

43. Regarding claims 18 and 21 rejected under 35 U.S.C. §103(a) as being unpatentable over Seal et al. (U.S. 6,965,984):

Appellant argues "Seal appears to teach a decoder 10 that can decode only ARM instructions. Moreover, Seals bytecode translation hardware operates only on Java Bytecodes. Thus, Seal fails to teach or fairly suggest decode logic that decodes instructions, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different from the first instruction set."

The examiner partially agrees for the following reasons. The examiner agrees with the appellant's assessment of Seal. However, the appellant has ignored the examiner's actual rejection for the claimed limitation. The examiner stated that it's obvious to one of ordinary skill in the art at the time of the invention that there would have to be some sort of predecoding done to initially detect java bytecodes to send to element 6 or detect ARM opcodes that will be bypassed around element 6. Thus, the appellant hasn't properly traversed the rejection of claim 18 and the examiner believes it therefore must be upheld.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/JAP/

/Jacob A. Petranek/

February 7, 2008

Conferees:

/Eddie P Chan/

Supervisory Patent Examiner, Art Unit 2183

Eddie Chan

Supervisory Patent Examiner
Technology Center 2100

/Lynne H Browne/
Lynne H Browne

Appeal Practice Specialist, TQAS
Technology Center 2100